

# A Bayesian Approach to the Data Description Problem

Alireza Ghasemi<sup>1</sup>

Hamid R. Rabiee<sup>2</sup>

Mohammad T. Manzuri<sup>2</sup>

M. H. Rohban<sup>2</sup>

alireza.ghasemi@epfl.ch rabiee@sharif.edu

manzuri@sharif.edu

rahban@ce.sharif.edu

<sup>1</sup>School of Computer and Communication Sciences  
Ecole Polytechnique Fédérale de Lausanne (EPFL)  
Lausanne, Switzerland

<sup>2</sup>AICTC Research Center  
Sharif University of Technology  
Azadi Avenue, Tehran, Iran

## Abstract

In this paper, we address the problem of data description using a Bayesian framework. The goal of data description is to draw a boundary around objects of a certain class of interest to discriminate that class from the rest of the feature space. Data description is also known as one-class learning and has a wide range of applications. The proposed approach uses a Bayesian framework to precisely compute the class boundary and therefore can utilize domain information in form of prior knowledge in the framework. It can also operate in the kernel space and therefore recognize arbitrary boundary shapes. Moreover, the proposed method can utilize unlabeled data in order to improve accuracy of discrimination.

We evaluate our method using various real-world datasets and compare it with other state of the art approaches of data description. Experiments show promising results and improved performance over other data description and one-class learning algorithms.

## Introduction

A critical assumption for many supervised learning algorithms is presence of training data from all classes under study. It means that, for example a binary classification algorithm requires training samples of both classes in order to work properly. In scenarios where this condition is not met, performance degrades considerably or even algorithm fails to run. A well-known example of such scenarios is the problem of image retrieval (with relevance feedback) in which the system is given only rare samples of the relevant class and therefore traditional supervised learning algorithms are not suitable for this problem.

The aforementioned problems are known as data description or one-class learning problems and have a wide range of applications from pattern recognition to data mining and image processing. Information retrieval, video surveillance, outlier detection and object detection are all among applications of one-class learning algorithms.

As well as presence of samples of only one class (which is called target class), there are also other scenarios in which

one-class learning can be beneficial. Another implicit assumption of many supervised learning algorithms is that the prior probabilities of different classes in the training set (and whole feature space) are equal or at least very close. However, this is also violated in many real-world situations such as spam detection in which the proportion of spam messages and regular e-mail is quite different in a fair data sample. One-class learning algorithms can also be beneficial in this case since they do not assume this and are designed for databases in which the proportion or other properties of different classes (like statistical distribution) are quite different. Examples of other problems of this kind are industrial fault detection and information retrieval.

Several one-class learning algorithms have been proposed so far. The work in (Khan and Madden 2010) is a recent survey on current trends in one-class learning. Many of these algorithms are extensions of traditional classification algorithms adapted to work in one-class settings. For example, in (Bishop 1994) an approach based on neural networks is proposed for novelty detection. Also in (Li and Zhang 2008) a variant of decision tree has been used for one-class learning. In (Yang et al. 2010) the  $k$  nearest neighbors algorithm has been used for one-class learning. Although such algorithms are simple and easy to understand, they are usually inefficient on complicated real-world data.

A major class of one-class learning algorithms are based on statistical density estimation. These approaches assume a parametric statistical model for the target class and then estimate the parameters of that model. The likelihood of a data sample measures the degree that the sample belongs to the target class. In (Cohen, Sax, and Geissbuhler 2008) approaches based on Parzen or kernel density estimation have been proposed. Also in (Nuez et al. 2009), Gaussian mixture models have been utilized for novelty detection. The principal advantage of these methods is the rigid theoretical foundations behind them. However, they can not directly operate in the kernel space and therefore have some limitations in modeling the complex boundary shapes.

Since the introduction of support vector machines (Burgess 1998) and kernel methods (Shawe-Taylor and Cristianini 2004), there has been a growing interest in adapting kernel-based approaches to one-class learning. Scholkopf in (Scholkopf et al. 2001) presented one-class SVM. It is a variation of traditional binary SVM which tries to sepa-

rate target data from the axis origin. (Tax and Duin 2004) proposed support vector data description. In this method, a hypersphere of minimum volume is sought to surround the target samples. In (Grnitz, Kloft, and Brefeld 2009), it is shown that the two approaches yield the same solution when the used kernel is isotropic. Kernel methods yield good results in most problem and model different kinds of boundary shapes utilizing flexibility of the kernel space. However, domain knowledge can not be easily embedded into kernel methods. Moreover, these methods can not directly utilize unlabeled data to improve accuracy.

Utilizing unlabeled data in the process of one-class learning has also been of interest in recent years. (Zhang and Zuo 2008), (Yu 2005) and (Elkan and Noto 2008) have utilized unlabeled samples as well as positive target samples in the process of one-class learning. These methods try to infer a set of confident negative samples from the unlabeled set and then perform a traditional binary classification algorithm. (Grnitz, Kloft, and Brefeld 2009) and (Tax and Duin 2004) have utilized outlier samples in addition to targets in the process of learning. The relation between support vector methods and density based approaches has been discussed in (Munoz and Moguerza 2004). In (Lee et al. 2007), the local density around target point has been used to improve the SVDD.

The Gaussian process regression method has been adapted for one-class learning in (Kemmler, Rodner, and Denzler 2011). Moreover, among the probabilistic approaches to one class learning, (Ting, D’Souza, and Schaal 2007) has used a Bayesian approach which defines a regression model over data samples. In (Coughlan and Yuille 2003; Varbanov 1998) Bayesian approaches have been used for outlier detection. In (Dong 2010) a Bayesian approach has been used to detect outliers in ordinal data. These methods are more flexible since they allow uncertainty in the model and use domain knowledge in constructing the classifier. However, their principal drawback is their computational inefficiency and lack of sparseness.

In this paper, we propose a novel Bayesian approach to the data description problem. The principal contribution of our work is twofold: First we develop a Bayesian framework which can benefit from advantages of both probabilistic and support vector approaches. For example our approach can generate sparse solutions and in addition, we propose a principled method for utilizing prior knowledge in the process of one-class learning. The second contribution of our work is that the proposed approach can benefit from unlabelled data in improving the accuracy of classification.

In the rest of this paper, after reviewing SVDD, a well-known data description algorithm, we propose our approach and study its properties and extensions. Then, we thoroughly test our approach against other one-class learning algorithms under various conditions.

## The Traditional Support Vector Data Description

Support vector data description is a well-known algorithm for one-class learning which has been widely used in various

applications. It is a kernel-based approach which tries to find a hypersphere which is as small as possible and meanwhile contains as much target data as possible, hereby avoiding outlier samples. This goal is achieved by solving a convex optimization problem over the target data points in the kernel space, in a way very similar to the well-known support vector machine algorithm.

We describe SVDD briefly in the rest of this section. For a more detailed explanation, refer to the seminal work of Tax and Duin.

Suppose we are given a dataset  $\{x_1, \dots, x_n\}$  which consists of the training set. The main idea of support vector data description is to find a hypersphere in the feature space containing as many of the training samples as possible while having minimum possible volume. To achieve this goal, data are first transformed to a higher dimensional kernel space in which support of the data is a hypersphere.

The sphere is characterized by its center  $C$  and radius  $R > 0$ . The minimization of the sphere volume is achieved by minimizing its square radius  $R^2$ . Data samples outside the hypersphere are penalized in the objective function. To consider the penalty, slack variables  $\xi_i \geq 0$  are introduced and the optimization problem is formulated as:

$$\min_{R \in \mathcal{R}, \xi_i \in \mathcal{R}^n, C \in \mathcal{F}} R^2 + \frac{1}{N\nu} \sum_{i=1}^N \xi_i \quad (1)$$

such that

$$\|\phi(x_i) - C\| \leq R^2 + \xi_i \text{ and } \xi_i \geq 0 \quad (2)$$

In the above formula,  $\phi$  is the transformation which maps data points to the higher dimensional. The parameter  $\nu$  controls the trade-off between the hypersphere volume and the proportion of samples in the hypersphere. It can also be used to control the sparseness of the solution of the optimization problem (Grnitz, Kloft, and Brefeld 2009).

Introducing Lagrangian multipliers to account for constraints, we obtain the following dual problem:

$$\min_{\alpha} \alpha^t \mathbf{K} \alpha - \alpha^t \text{diag}(\mathbf{K}) \quad (3)$$

such that

$$0 \leq \alpha_i \leq \frac{1}{N\nu} \text{ and } \sum \alpha_i = 1 \quad (4)$$

In (3),  $\mathbf{K}$  is the kernel matrix in which  $\mathbf{K}_{i,j} = \langle \phi(x_i), \phi(x_j) \rangle$  and  $\text{diag}(\mathbf{K})$  is the main diagonal of  $\mathbf{K}$ . One may notice that it is not needed to explicitly transform data to the kernel space and only defining a kernel function (dot product between transformed data) in terms of original points is sufficient. We call this function  $\mathcal{K}(\cdot, \cdot)$ . Therefore,  $\mathcal{K}(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$ .

Solving the dual optimization problem yields vector  $\alpha$  in which most of the values are 0. Samples  $x_i$  with positive  $\alpha_i$  are called support vectors of the SVDD. center  $C$  of the hypersphere can be specified in term of Lagrange multipliers  $\alpha_i$  as:

$$C = \sum_i \alpha_i \phi(x_i) \quad (5)$$

We can rank test samples by their proximity to the center of the hyper sphere. The ranking function  $f$  is defined as below in which smaller values of  $f$  mean more similarity to the target class.

$$f(z) = \sum_i \sum_j \alpha_i \alpha_j \mathcal{K}(x_i, x_j) + \mathcal{K}(z, z) - 2 \sum_i \alpha_i \mathcal{K}(x_i, z) \quad (6)$$

## The Bayesian Approach

As we saw in the previous section, the support vector data description algorithm finally reduces to finding center of the surrounding hypersphere in the embedded space as a weighted average of sample target points in which many of the weights are zero. Data points for which the corresponding weight is non-zeros are called support vectors.

In this section we derive the proposed Bayesian data description method. We look at the problem of data description from a different point of view. Later we show that the interpretation of data and parameters in our model is equivalent to that of SVDD.

Our method is based on the same set of parameters as the SVDD (in its dual form), i.e we will try to find a vector of weights, one for each data sample. Assume that we transform all data samples using the mapping  $\phi$  to a higher dimensional embedded (kernel) space in which transformed data follow a Gaussian distribution with covariance matrix  $I$  and mean  $\sum_i \alpha_i \phi(x_i)$ . i.e.:

$$\phi(x_j) \sim \mathcal{N}(\sum_i \alpha_i \phi(x_i), I) \quad (7)$$

We limit  $\alpha_i$  values to form a convex set, i.e.  $0 < \alpha_i < 1$  and  $\sum_i \alpha_i = 1$ . Later, we will discuss the reason behind this assumption.

The main difference between the estimation in (7) and the conventional Gaussian density estimation is that the mean is limited to be a weighted average of training target points. Hereafter, we call this model the weighted Gaussian model.

The principal correspondence between the weighted Gaussian model and the SVDD is that the mean of the weighted Gaussian is equivalent to the center of the hypersphere in the SVDD. Therefore, distance of a point to center of the surrounding hypersphere in the SVDD model is inversely proportional to the likelihood of a data point in the weighted Gaussian model. We use this fact to show that SVDD is itself a special case of the weighted Gaussian model. Then we improve upon SVDD equivalent case of weighted Gaussian by utilizing unlabeled data and defining more precise prior knowledge.

To achieve this goal, first we have to estimate parameters of the weighted Gaussian model using a statistical parameter estimation approach. Various parameter estimation methods have been proposed in the literature so far. Two of the most common ones are maximum likelihood ones and Bayesian approach.

Maximum likelihood estimation is a simple optimization-based approach which maximizes the likelihood of training data with regard to the unknown parameters. However,

this method is not flexible and can not utilize domain information to improve the estimation. We seek to arbitrarily constrain the estimation procedure toward finding solution with specific properties (e.g. sparseness) and moreover utilize various forms of domain information in this procedure. Therefore, we use the Bayesian estimation.

In Bayesian parameter estimation, a prior distribution  $p(\alpha)$  is defined over parameter vector  $\alpha$  and the posterior probability  $p(\alpha|D)$  is derived by applying the Bayes rule:

$$p(\alpha|D) = \frac{p(D|\alpha)p(\alpha)}{p(D)} \quad (8)$$

In which  $p(D|\alpha)$  is the likelihood of training data given a specific value of  $\alpha$  and  $p(D)$  is a normalizing constant.

We assume that the parameter vector  $\alpha$  follows a Gaussian distribution with mean  $m$  and covariance matrix  $C$  i.e.:

$$\alpha \sim \mathcal{N}(m, C) \quad (9)$$

Applying the Bayes rule, we have:

$$p(\alpha|D) \propto p(D|\alpha)p(\alpha) \quad (10)$$

We have omitted  $p(D)$  in (10) because it is independent of  $\alpha$ .

Maximizing the a posteriori probability of  $\alpha$  (MAP estimation) we will have:

$$\hat{\alpha} = \arg \min_{\alpha} \alpha^t (n\mathbf{K} + C^{-1})\alpha - 2\alpha^t (\mathbf{D}\mathbf{1} + C^{-1}m) \quad (11)$$

Matrix  $\mathbf{D}$  is the diagonal matrix of weighted degree of samples, i.e.  $\mathbf{D}_{i,i} = \sum_j \mathbf{K}_{i,j}$ .

Equation (11) is the key to our approach since it allows purposeful modification of the behavior of the final solution by setting different values for covariance matrix  $C$  and mean  $m$  of the parameter vector. For example, the objective function of SVDD can be derived from (11) by choosing the appropriate  $C$  and  $m$  (We can check this by substituting  $C = I$  and  $m = \text{diag}(\mathbf{K}) - \mathbf{D}\mathbf{1}$  and assuming an isotropic Kernel). Moreover, we see that the optimization only depends on dot products of points in the embedded (kernel) space. Therefore, the Bayesian estimation to the weighted Gaussian model is itself a kernel method. That is why we constrained the mean of the model to be a weighted average of training points.

The most trivial choice for the parameters could be setting  $C$  to the identity matrix and each  $m_i$  equally to  $\frac{1}{n}$ . However, this kind of prior knowledge is non-informative and therefore yields the same non-sparse trivial solution as the maximum likelihood approach.

As a better and more informative prior knowledge, we could modify the mean vector  $m$  such that the data points which lie in a dense area of the embedded space receive smaller prior weight. The main motivation behind this choice is the fact that target points located in dense areas of feature space are less likely to be close to boundary of the target class and therefore their corresponding weight

**Require:** Set of Target Training Samples  $T$   
**Require:** Set of Target Test Samples  $S$   
1: Compute Kernel matrix  $\mathbf{K}$  from training data  
2: Compute diagonal matrix  $\mathbf{D}$  such that  $D_{ii} = \sum_j \mathbf{K}_{i,j}$   
3:  $n \leftarrow \text{number of training samples}$   
4:  $n_{test} \leftarrow \text{number of test samples}$   
5:  $\mathbf{C} \leftarrow \mathbf{I}_{n \times n}$   
6:  $\forall i : 1 \leq i \leq n \rightarrow m_i = -(\sum_j \mathbf{K}_{i,j})^\nu$   
7:  $\alpha = \arg \min_{\alpha} \alpha^t (n\mathbf{K} + \mathbf{C}^{-1})\alpha - 2\alpha^t (\mathbf{D}\mathbf{1} + \mathbf{C}^{-1}\mathbf{m})$   
8: **for**  $i = 1 \rightarrow n_{test}$  **do**  
9:    $score_i = \sum_j \Sigma_k \alpha_j \alpha_k \mathcal{K}(x_j, x_k) + \mathcal{K}(x_i, x_i) - 2\sum_j \mathcal{K}(x_i, x_j)$   
10: **end for**  
11: Sort test samples in ascending order by the  $score$  values  
12: **return** Desired number of samples from top of the list

Figure 1: Bayesian data description (BDD) algorithm

should have more prior probability of being close to zero. In contrast, target points located in less-dense areas of feature space are more likely to be on or close to the boundary and therefore their corresponding weight should be a priori larger than other points.

With these facts in mind, we suggest that the mean of prior probability of parameter vector be proportional to  $\mathbf{D}\mathbf{1}$ , in which  $\mathbf{D}$  is the same diagonal matrix as in (11). This is reasonable because the weighted degree of a point is a good approximation of local density of the area near that point. Therefore we set:

$$m_i \propto -\sum_{j \in L} \mathbf{K}_{i,j} \quad (12)$$

for each element of mean vector  $\mathbf{m}$ . Using such prior knowledge, we expect that samples crucial in determining center of the Gaussian become much more likely to receive larger values. This causes the solution to become sparse and more accurately capture the underlying distribution and its support (boundary).

The pseudo code for the Bayesian data description algorithm is depicted in figure 1. In this algorithm, parameter  $0 < \nu < 1$  controls sparsity of the solution. Larger values for  $\nu$  cause the solution to become as sparse as possible.  $\nu$  can also be used to reduce the effect of outlier data on the final solution.

Also figure 2 depicts performance of our weighted Gaussian model (with maximum likelihood and Bayesian estimation) in capturing a typical S-shaped distribution and compares it with that of SVDD. We see that weighted Gaussian with maximum likelihood estimation has captured a mostly spherical distribution shape which shows that this method lacks sparsity and flexibility and its solution is close to the simple mean of points which is the trivial solution. Both BDD and SVDD has been more successful in capturing the real shape of the distribution and avoid over-fitting.

## Utilizing Unlabelled Data

Methods which utilize unlabeled data to improve learning accuracy have received much attention in recent years. These methods use unlabeled data to infer information about geometry of data and the corresponding manifold. Such information can be used to improve the accuracy of supervised classifiers.

In the Bayesian data description approach, information about the geometry of data can be utilized to determine the prior probability distribution of the parameter vector. Since we use local density of area around points in determining the prior probability distribution of the model parameters. The information we gather from unlabeled samples can be useful in determining the local density around a point, more accurately. Having unlabeled data available, we can now set:

$$m_i = -\sum_{j \in L \cup U} \mathbf{K}_{i,j} \quad (13)$$

In which  $L$  and  $U$  are the set of labeled and unlabeled data, respectively.

Another parameter which can be modified by using unlabeled data is the covariance matrix  $\mathbf{C}$ . Information about geometry of data can be used in constructing this matrix by using any type of data dependent kernel.

An example of using unlabeled data for adjusting the covariance is by computing the Laplacian operator of training samples (Zhu, Lafferty, and Rosenfeld 2005).

Suppose we define a  $k$ -nn graph over all data samples. A  $k$ -nn graph is a graph in which nodes are data samples and each sample is connected to its  $k$  nearest neighbors. Weight  $W_{i,j}$  of each edge is proportional to the similarity between data samples  $x_i$  and  $x_j$ . Gaussian function is a popular choice for  $W$ .

Having the weight matrix, the Laplacian  $L$  of the graph is defined as  $L = \mathbf{D} - \mathbf{W}$  in which  $\mathbf{D}$  is a diagonal matrix in which  $D_{i,i} = \sum_j W_{i,j}$ . Utilizing the Laplacian, we adjust matrix  $\mathbf{C}$  as:

$$\mathbf{C}^{-1} = (\mathbf{L}^{-1})_{1 \dots n, 1 \dots n} \quad (14)$$

Utilizing unlabeled data in this way, manifold of all data (target and non-target) is modeled in the  $\mathbf{C}$  matrix, whereas manifold of target data can be modeled in the kernel matrix  $\mathbf{K}$  of the weighted Gaussian itself. Therefore, we use both manifolds to better model distribution of the data.

## Time Complexity of the Bayesian Data Description

Constructing the prior vector  $\mathbf{m}$  can be done at the time of constructing the kernel matrix and requires  $\mathcal{O}(n^2)$ , the same as minimum complexity of kernel construction (in the general sense). The objective function of the BDD is a convex quadratic programming problem which can be solved in  $\mathcal{O}(n^3)$  time. SVDD also reduces to a quadratic programming problem. Therefore the time complexity of BDD is not higher than SVDD.

In the semi-supervised settings (SSDD), we require to compute inverse of the covariance matrix which is of complexity  $\mathcal{O}((n+m)^3)$  ( $m$  is the number of unlabeled samples). The prior weight vector can still be formed at the time of kernel construction with the same complexity as kernel construction ( $\mathcal{O}((n+m)^2)$ ). Finally, the resulting quadratic requires  $\mathcal{O}(n^3)$  time to be solved which is independent of the number of unlabeled data.

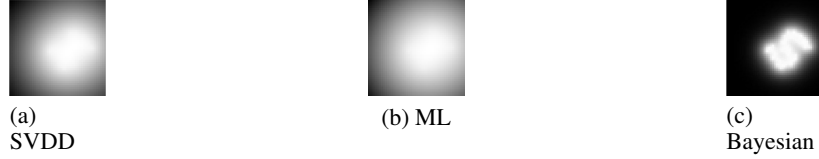


Figure 2: Performance of density estimation for SVDD against the proposed weighted Gaussian

Table 1: Datasets Used in Experiments

Dataset	No. of features	No. of Samples	No. of Classes
Iris	256	150	3
USPS	256	9998	10
Pendigits	16	10992	10
ISOLET	617	7797	10
MNIST	784	60000	10
COIL20	1024	1440	20
Caltech-101	144	9144	101
Letter	16	1259	26
Glass	9	214	6
Corel	144	1000	20

## Experimental Results

### Experiment Setup

Various datasets from the UCI repository (Asuncion and Newman 2007), as well as Corel (Wang, Li, and Wiederhold 2001) and Caltech-101 (Fei-Fei, Fergus, and Perona 2007) image databases were used for experiments. Their properties are depicted in table 1.

In each experiment, one of the classes was selected as target, and all other samples were treated as outlier. One-half of the target samples were selected for training. The rest of training samples, as well as outlier data were selected as test samples. For the Caltech-101 and Corel image datasets, feature extraction was performed by the CEDD (Chatzichristofis and Boutalis 2008) feature extraction algorithm.

SVDD method and one-class Gaussian process were implemented and compared with the proposed BDD method. The Gaussian function was used as the kernel. The parameters of the classifiers and the kernel were adjusted by 10-fold cross validation. All sample selections were done by random sampling.

For measuring efficiency of one-class learning, we computed precision in the top  $k$  returned results as accuracy measure and set  $k$  to the (estimated) number of target samples in the test set. This measure has the advantage that unlike precision or recall, we don't need to compute more than one quantity in order to achieve meaningful results. Moreover, the value chosen for  $k$  eliminates the need for explicitly adjusting an acceptance threshold for one-class learning algorithms which could be a tedious task and have significant effect on functionality of algorithms.

### Experiments

In table 2 we compare the performance of BDD with that of SVDD and one-class Gaussian process. The BDD and

SVDD show similar performance with slight improvements in BDD because of utilizing the density-based prior knowledge. One-class Gaussian process also has a reasonable performance but this algorithm is not sparse and therefore lacks benefits of the other models and is more hardly generalizable. Running times (in seconds) of algorithms are depicted in parentheses in each cell.

Figure 3 shows interesting results about performance of the Bayesian data description on different classes of the USPS digit recognition dataset. Here, we visualize different data samples in order to understand the operation of BDD. Each column depicts performance on one class of the USPS dataset.

The first row shows the most likely samples of each class returned by the BDD algorithm. As can be seen, all samples in this row have been classified correctly and are appropriate representatives for their respective class.

The second row shows the least likely sample detected as target by the BDD for each class. We can see that these samples are misclassified data and count as error rate of the classifier. It is reasonable to have error here since we rank data samples by likelihood to the target class and samples with lower ranks are more likely to be misclassified (unless the precision is perfect 1).

The third and fourth row deal with the prior estimation of the local density around each sample which is done by computing its weighted degree. The third row shows the data sample with least weighted degree. We see that these samples usually can not be considered typical representatives of their underlying target class and should be far from the center of mass of the target class. These data samples lie in the boundary of target class and therefore have the most important role in defining the center of the weighted Gaussian model. Because of this property of weighted degree of data samples, we set the prior probability of the parameter corresponding to each sample proportional to the weighted degree of that sample.

The fourth row shows the sample with largest weighted degree. We can see that the data samples are typical representatives of their underlying class. This is because samples with large weighted degree usually lie within the target hypersphere and are far from the boundary of the target class.

An important point to note about one-class learning algorithms is their sensitivity to the proportion of target and outlier data samples in the test set. The accuracy of the resulting model can be affected significantly by varying this ratio. We test this by gradually increasing the proportion of outlier samples in the test set and computing precision in each case. Figure 4 depicts results of studying this property



Figure 3: Performance of BDD on different classes of the USPS dataset

Table 2: Experimental results with supervised Bayesian data description and other one-class learning algorithms

Dataset	OCGP	SVDD	BDD
Iris	$97.85 \pm 0.13(0.82)$	$98.08 \pm 0.08(0.54)$	<b><math>98.11 \pm 0.03(0.35)</math></b>
USPS	$89.22 \pm 0.04(2.10)$	$89.14 \pm 0.03(1.29)$	<b><math>89.23 \pm 0.03(1.43)</math></b>
Pendigits	$95.75 \pm 0.22(1.92)$	$94.65 \pm 0.10(1.61)$	<b><math>95.91 \pm 0.12(1.54)</math></b>
ISOLET	$91.28 \pm 0.87(2.31)$	$92.37 \pm 0.51(1.21)$	<b><math>94.52 \pm 0.54(1.24)</math></b>
MNIST	$87.46 \pm 0.92(3.86)$	$85.01 \pm 0.30(3.82)$	<b><math>88.51 \pm 0.34(3.95)</math></b>
COIL20	$51.33 \pm 3.04(2.32)$	<b><math>58.54 \pm 1.87(1.74)</math></b>	$57.01 \pm 2.38(1.79)$
Caltech-101	$79.83 \pm 1.02(2.14)$	$80.07 \pm 0.91(2.18)$	<b><math>82.21 \pm 0.60(2.01)</math></b>
Letter	<b><math>83.10 \pm 1.19(1.14)</math></b>	$80.23 \pm 0.81(0.93)$	$82.41 \pm 0.94(0.91)$
Glass	$77.91 \pm 1.81(0.14)$	$77.12 \pm 1.70(0.09)$	<b><math>79.34 \pm 1.72(0.09)</math></b>
Corel	$92.21 \pm 1.51(1.58)$	$90.16 \pm 1.17(1.41)$	<b><math>93.19 \pm 1.19(1.41)</math></b>

for SVDD and BDD model.

As can be seen in figure 4a, in large datasets the precision of classification is not affected largely by increasing the proportion of outlier samples. This is mostly because the training set is big enough to capture the distribution of target class. Presence of sufficient target samples prevents probably noisy data to affect misclassification rate.

However, figure 4b depicts that this is not the case for smaller datasets like Corel and Caltech-101. Here, due to insufficiency of target training samples, noisy data can significantly influence the boundary of target class and hence misclassification rate increases by increasing the proportion of outlier samples.

We can see in both figures that BDD is less sensitive to variations in the proportion of outlier samples, which is mostly because of its use of prior knowledge over model parameters. By using weighted degree as a prior, we prevent noisy data to become significant in constructing the model and compensate for the model uncertainty.

**Experiments with unlabeled data** For semi-supervised learning, we divided the training set into a labeled and an unlabeled set. We set the size of unlabeled set twice the size of labeled training set and for better runtime, used the unlabeled data only to improve the prior mean of Bayesian model. In addition to SVDD, the mapping-convergence algorithm (Yu 2005) was also implemented and used in comparisons. Results of semi-supervised learning are depicted in figure 3.

We see in table 3 that semi-supervised Bayesian data description algorithm (SSDD) outperforms other approaches. Since SVDD can not use unlabeled data, it is expectable that we don't see any performance improvement by adding unlabeled data. Mapping-convergence also does not achieve good performance, because this algorithm uses unlabeled data only to select some confident negative samples and then performs a traditional binary classification algorithm. There-

Table 3: Experimental results with semi-supervised Bayesian data description learning algorithms.

Dataset	SVDD	MC	SSDD
Iris	$98.06 \pm 0.09(0.52)$	$98.17 \pm 0.04(1.54)$	<b><math>99.89 \pm 0.01(0.75)</math></b>
USPS	$89.19 \pm 0.04(1.30)$	$88.23 \pm 0.02(2.57)$	<b><math>94.76 \pm 0.05(2.16)</math></b>
Pendigits	$94.75 \pm 0.12(1.72)$	$96.01 \pm 0.07(2.70)$	<b><math>98.89 \pm 0.10(1.98)</math></b>
ISOLET	$92.28 \pm 0.57(1.22)$	$94.87 \pm 0.23(2.60)$	<b><math>98.23 \pm 0.38(2.02)</math></b>
MNIST	$85.06 \pm 0.32(3.81)$	$90.01 \pm 0.18(9.82)$	<b><math>94.48 \pm 0.31(5.07)</math></b>
COIL20	$54.63 \pm 2.00(1.74)$	$59.25 \pm 1.06(4.46)$	<b><math>66.53 \pm 2.51(2.75)</math></b>
Caltech-101	$80.01 \pm 0.86(2.14)$	$83.07 \pm 0.31(4.78)$	<b><math>89.90 \pm 0.58(3.21)</math></b>
Letter	$80.20 \pm 0.91(1.00)$	$88.34 \pm 0.30(2.06)$	<b><math>95.10 \pm 0.79(1.61)</math></b>
Glass	$77.20 \pm 1.71(0.10)$	$79.12 \pm 1.01(1.00)$	<b><math>86.02 \pm 1.61(0.86)</math></b>
Corel	$90.21 \pm 1.21(1.48)$	$93.69 \pm 0.89(3.41)$	<b><math>97.26 \pm 1.24(2.52)</math></b>

fore, the problems that arise for binary classification on one-class problems also arises for this algorithm and degrades its performance. Moreover, we can see that smaller and more difficult datasets are improved more significantly by utilizing unlabeled data. This is because of the fact that the training data are insufficient for these problems and therefore they benefit more from using the unlabeled data.

Also running time (in seconds) of each algorithm is depicted in parentheses in table 3. We can see that SSDD performs quite faster than mapping-convergence and also it's speed is very close to that of SVDD that does not use unlabeled data at all. The mapping-convergence algorithm is slower than SSDD because it runs both one-class learning (to detect negative points) and a traditional binary classification, whereas SSDD only runs data description.

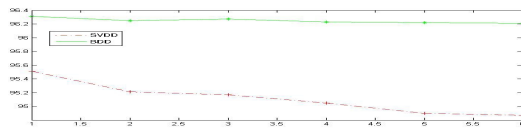
## Conclusions

In this paper, we proposed a novel Bayesian approach for the data description problem which has various applications in machine learning. Our approach is a bridge between probabilistic and kernel based data description and hence can use benefits of both types of approaches such as sparseness of the support vector approaches and utilizing prior knowledge in the probabilistic approaches. Moreover, our approach can utilize unlabeled data in order to improve accuracy of the data description.

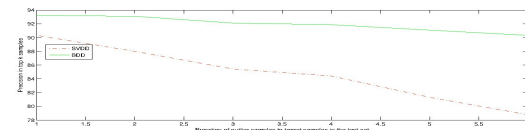
The prior knowledge utilized in our model can have various applications. For example, the information in the data samples prior, can be used to estimate most probable support vectors and reduce the size of data set, hereby reducing time complexity of the training. Moreover, robustness of the algorithm to noise could be further improved.

## Acknowledgment

The authors thank the AICTC research center and VAS Laboratory of Sharif University of Technology.



(a) Pendigits



(b) Corel

Figure 4: Precision against proportion of target and outlier samples for big and small datasets

## References

- [Asuncion and Newman 2007] Asuncion, A., and Newman, D. 2007. Uci machine learning repository.
- [Bishop 1994] Bishop, C. 1994. Novelty detection and neural network validation. In *Vision, Image and Signal Processing, IEE Proceedings-*, volume 141, 217–222. IET.
- [Burges 1998] Burges, C. 1998. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery* 2(2):121–167.
- [Chatzichristofis and Boutalis 2008] Chatzichristofis, S., and Boutalis, Y. 2008. Cedd: Color and edge directivity descriptor: A compact descriptor for image indexing and retrieval. In *Proceedings of the 6th international conference on Computer vision systems*, 312–322. Springer-Verlag.
- [Cohen, Sax, and Geissbuhler 2008] Cohen, G.; Sax, H.; and Geissbuhler, A. 2008. Novelty detection using one-class parzen density estimator. an application to surveillance of nosocomial infections. In *EHealth Beyond the Horizon: Get It There: Proceedings of MIE2008 the XXIst International Congress of the European Federation for Medical Informatics*, 21. Ios Pr Inc.
- [Coughlan and Yuille 2003] Coughlan, J., and Yuille, A. 2003. Manhattan world: Orientation and outlier detection by bayesian inference. *Neural Computation* 15(5):1063–1088.
- [Dong 2010] Dong, F. 2010. Bayesian method to detect outliers for ordinal data. *Communications in Statistics-Simulation and Computation* 39(7):1470–1484.
- [Elkan and Noto 2008] Elkan, C., and Noto, K. 2008. Learning classifiers from only positive and unlabeled data. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 213–220. ACM.
- [Fei-Fei, Fergus, and Perona 2007] Fei-Fei, L.; Fergus, R.; and Perona, P. 2007. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding* 106(1):59–70.
- [Grnitz, Kloft, and Brefeld 2009] Grnitz, N.; Kloft, M.; and Brefeld, U. 2009. Active and semi-supervised data domain description. *Machine Learning and Knowledge Discovery in Databases* 407–422.
- [Kemmler, Rodner, and Denzler 2011] Kemmler, M.; Rodner, E.; and Denzler, J. 2011. One-class classification with gaussian processes. *Computer Vision-ACCV 2010* 489–500.
- [Khan and Madden 2010] Khan, S., and Madden, M. 2010. A survey of recent trends in one class classification. *Artificial Intelligence and Cognitive Science* 188–197.
- [Lee et al. 2007] Lee, K.; Kim, D.; Lee, K.; and Lee, D. 2007. Density-induced support vector data description. *Neural Networks, IEEE Transactions on* 18(1):284–289.
- [Li and Zhang 2008] Li, C., and Zhang, Y. 2008. Bagging one-class decision trees. In *Fifth International Conference on Fuzzy Systems and Knowledge Discovery*, 420–423. IEEE.
- [Munoz and Moguerza 2004] Munoz, A., and Moguerza, J. 2004. One-class support vector machines and density estimation: The precise relation. *Progress in Pattern Recognition, Image Analysis and Applications* 253–274.
- [Nuez et al. 2009] Nuez, P.; Drews, P.; Rocha, R.; Campos, M.; and Dias, J. 2009. Novelty detection and 3d shape retrieval based on gaussian mixture models for autonomous surveillance robotics. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, 4724–4730. IEEE.
- [Schlkopf et al. 2001] Schlkopf, B.; Platt, J.; Shawe-Taylor, J.; Smola, A.; and Williamson, R. 2001. Estimating the support of a high-dimensional distribution. *Neural computation* 13(7):1443–1471.
- [Shawe-Taylor and Cristianini 2004] Shawe-Taylor, J., and Cristianini, N. 2004. *Kernel methods for pattern analysis*. Cambridge Univ Pr.
- [Tax and Duin 2004] Tax, D., and Duin, R. 2004. Support vector data description. *Machine learning* 54(1):45–66.
- [Ting, D’Souza, and Schaal 2007] Ting, J.; D’Souza, A.; and Schaal, S. 2007. Automatic outlier detection: A bayesian approach. In *Robotics and Automation, 2007 IEEE International Conference on*, 2489–2494. IEEE.
- [Varbanov 1998] Varbanov, A. 1998. Bayesian approach to outlier detection in multivariate normal samples and linear models. *Communications in Statistics-Theory and Methods* 27(3):547–557.
- [Wang, Li, and Wiederhold 2001] Wang, J. Z.; Li, J.; and Wiederhold, G. 2001. Simplicity: Semantics-sensitive integrated matching for picture libraries. *IEEE Trans. Pattern Anal. Mach. Intell.* 23(9):947–963.
- [Yang et al. 2010] Yang, J.; Zhong, N.; Liang, P.; Wang, J.; Yao, Y.; and Lu, S. 2010. Brain activation detection by neighborhood one-class svm. *Cognitive Systems Research* 11(1):16–24.
- [Yu 2005] Yu, H. 2005. Single-class classification with mapping convergence. *Mach. Learn.* 61:49–69.
- [Zhang and Zuo 2008] Zhang, B., and Zuo, W. 2008. Learning from positive and unlabeled examples: A survey. In

*Information Processing (ISIP), 2008 International Symposiums on*, 650–654. IEEE.

[Zhu, Lafferty, and Rosenfeld 2005] Zhu, X.; Lafferty, J.; and Rosenfeld, R. 2005. *Semi-supervised learning with graphs*. Citeseer.